# 近似GCD算法GPGCDの複素係数多項式への拡張 (Computer Algebra : Design of Algorithms, Implementations and Applications)

AUTHOR(S):

照井, 章

CITATION:

照井, 章. 近似GCD算法GPGCDの複素係数多項式への拡張 (Computer Algebra : Design of Algorithms, Implementations and Applications). 数理解析研究所講究録 2012, 1814: 97-107

ISSUE DATE:

2012-10

URL:

http://hdl.handle.net/2433/194547

数理解析研究所講究録
第1814巻 2012年 97-107



# 近似GCD算法GPGCDの複素係数多項式への拡張
# GPGCD, an iterative method for calculating approximate GCD of univariate polynomials, with the complex coefficients

照井 章*
AKIRA TERUI
筑波大学大学院 数理物質科学研究科
GRADUATE SCHOOL OF PURE AND APPLIED SCIENCES, UNIVERSITY OF TSUKUBA

**Abstract**

我々は, これまでに, 1変数多項式に対する近似GCDの反復算法であるGPGCD法を提案している. これは, 与えられた多項式および次数に対し, 可能な限り小さな摂動を加えることにより, 与えられた次数のGCDおよびそのための摂動を求める算法である. GPGCD算法は, 与えられた問題を制約つき最小化問題に帰着させ, これを, 勾配射影法の一般化の一つである「修正Newton法」と呼ばれる反復算法で解くものである. 本稿では, GPGCD算法の入力多項式を2個の複素係数1変数多項式に対応させた拡張を提案する.

**Abstract**

We present an extension of our GPGCD method, an iterative method for calculating approximate greatest common divisor (GCD) of univariate polynomials, to polynomials with the complex coefficients. For a given pair of polynomials and a degree, our algorithm finds a pair of polynomials which has a GCD of the given degree and whose coefficients are perturbed from those in the original inputs, making the perturbations as small as possible, along with the GCD. In our GPGCD method, the problem of approximate GCD is transfered to a constrained minimization problem, then solved with a so-called modified Newton method, which is a generalization of the gradient-projection method, by searching the solution iteratively. While our original method is designed for polynomials with the real coefficients, we extend it to accept polynomials with the complex coefficients in this paper.

## 1 はじめに

多項式や行列を対象とする代数計算（数式処理）において, 数式・数値融合算法は, 最近, 注目を集めている. 中でも, 近似代数計算, すなわち, 与えられた問題自体には代数的関係が存在しないが, その近傍に, 代数的関係をもつものが存在した場合に, そのような代数的関係をもつ系を, もとの系からの摂動をなるべく小さく保ちながら探索するような計算は, 従来の計算代数の算法が適用不可能もしくは困難であるような, 浮動小数係数多項式などの問題に, 計算代数的手法を適用可能なものとして, 期待されている.

本稿では, 近似代数計算 [20] の算法として, 近似最大公約子 (GCD) の問題を取り上げる. これは, 多項式の組（一般的には互いに素）と, 次数$d$が与えられたときに, 与えられた多項式の係数に摂動を加え, $d$次のGCDをもつような系を探索し, 見つかったGCDを, 与えられた多項式の近似GCDと呼ぶものである.

*terui@math.tsukuba.ac.jp

近似GCDは, 近似代数計算の中でも最も古くから活発に研究が行われてきた問題の一つで, これまでに様々な算法が提案されており, それらの中には, 多項式剰余列 (PRS) に基づく算法 ([1], [12], [13]), Sylvesterの終結式行列の特異値分解 (SVD) に基づく算法 ([3], [6]), Sylvesterの終結式行列のQR分解に基づく算法 ([4], [17], [19]), Padé近似に基づく算法 [9], 最適化法に基づく算法 (下記参照) などがある. さらに, 種々の悪条件問題に対する解法 ([4], [11], [21]) も議論されている.

本稿では, これらの中で, 近似GCDの問題を制約つき最小化問題に帰着させ, 反復解法で解く最適化法に着目する. 最適化法を用いた既存の算法では, Levenberg-Marquard法 [2] やGauss-Newton法 [18], 構造化行列を用いた最小二乗法の一種であるSTLN法 (Structured Total Least Norm) ([8], [7]) 等が用いられており, 特に, 最近提案されたSTLN法に基づく算法は, 近似GCDを得るのに要する係数の摂動を小さく抑える点で注目を集めている.

我々は, これまでの研究で, GPGCDと呼ばれる反復算法を提案した [16]. これは, 与えられたGCDの計算問題を制約つき最適化問題に帰着させ, 勾配射影法の一般化と位置づけることのできる, 田邊 ([14], [22, 第4章]) による修正Newton法を用いて局所的最小解を探索するものである. 実験結果では, STLN法に基づく算法と同程度の摂動で近似GCDを探索でき, かつ大幅な効率化が図られることを示した. これまでに提案した算法は, 2つの実係数多項式の近似GCDを計算するものであったが, 本稿では, GPGCD算法を2つの複素係数多項式に対応させた拡張を提案する.

## 2 近似GCD問題と制約つき最小化問題への帰着

$F(x), G(x)$ を複素係数1変数多項式の組とし, 次式で与えられるものとする.

$$F(x) = x^m + f_{m-1}x^{m-1} + \cdots + f_0, \quad G(x) = x^n + g_{n-1}x^{n-1} + \cdots + g_0, \tag{1}$$

ここに, $m \geq n > 0$ とし, $F$ と $G$ は一般に互いに素であるとする. 与えられた次数 $d$ (ただし $n \geq d > 0$) に対し, $F(x)$ と $G(x)$ の係数に摂動を加えることにより, 次式のような $\tilde{F}(x)$ と $\tilde{G}(x)$ を計算することを考える.

$$\tilde{F}(x) = F(x) + \Delta F(x) = H(x) \cdot \bar{F}(x), \quad \tilde{G}(x) = G(x) + \Delta G(x) = H(x) \cdot \bar{G}(x). \tag{2}$$

ここに, $\Delta F(x)$, $\Delta G(x)$ は, 一般に複素係数多項式で, 次数がそれぞれ $F(x)$, $G(x)$ の次数を超えないような多項式, $H(x)$ は $d$ 次の (一般に複素係数) 多項式で, $\bar{F}(x)$ と $\bar{G}(x)$ は互いに素とする. 式 (2) をみたす $\tilde{F}$, $\tilde{G}$, $\bar{F}$, $\bar{G}$, $H$ が計算されたとき, $H$ を $F$ と $G$ の**近似GCD**と呼ぶ. 本稿では, **与えられた次数 $d$ に対し, 摂動のノルム $\|\Delta F(x)\|_2^2 + \|\Delta G(x)\|_2^2$ をなるべく小さく保ちつつ, $F$ と $G$ の $d$ 次の近似GCD $H$ を探索する問題を解く**.

$\tilde{F}(x)$ と $\tilde{G}(x)$ の $k$ 次部分終結式を $\mathrm{S}_k(\tilde{F}, \tilde{G})$ で表す. 多項式 $\tilde{F}(x)$ と $\tilde{G}(x)$ が $d$ 次のGCDをもつとき, 部分終結式の理論より, $\tilde{F}$ と $\tilde{G}$ の $d-1$ 次部分終結式は0に等しくなる, すなわち

$$\mathrm{S}_{d-1}(\tilde{F}, \tilde{G}) = 0$$

が成り立つ. このとき, $\tilde{F}$ と $\tilde{G}$ の $d-1$ 次部分終結式行列

$$N_{d-1}(F,G)=\begin{pmatrix} f_m & & & g_n & & \\ \vdots & \ddots & & \vdots & \ddots & \\ f_0 & & f_m & g_0 & & g_n \\ & \ddots & \vdots & & \ddots & \vdots \\ & & f_0 & & & g_0 \end{pmatrix}, \tag{3}$$

$$\underbrace{\qquad\qquad}_{n-d+1} \quad \underbrace{\qquad\qquad}_{m-d+1}$$

(式 (3) のように, $\tilde{F}$ の係数を $n-d+1$ 列, $\tilde{G}$ の係数を $m-d+1$ 列並べた行列) ランクが full rank から 1 落ちるため, 互いに素な多項式 $A(x)$ と $B(x)$ が存在して

$$A\tilde{F}+B\tilde{G}=0 \tag{4}$$

(ただし $\deg(A)<n-d$, $\deg(B)<m-d$) をみたす. ゆえに, 本稿で考える問題は, 与えられた $F(x)$, $G(x)$, $d$ に対し, 方程式 (4) をみたすような $\Delta F(x)$, $\Delta G(x)$, $A(x)$, $B(x)$ で, $\|\Delta F\|_2^2+\|\Delta G\|_2^2$ がなるべく小さくなるものを探索する問題に帰着される.

以下では, $F(x)$, $G(x)$ を

$$\begin{aligned} F(x)&=(f_{m,1}+f_{m,2}\boldsymbol{i})x^m+\cdots+(f_{0,1}+f_{0,2}\boldsymbol{i}),\\ G(x)&=(g_{n,1}+g_{n,2}\boldsymbol{i})x^n++\cdots+(g_{0,1}+g_{0,2}\boldsymbol{i}), \end{aligned} \tag{5}$$

で表す. ここに, $f_{j,1}$, $g_{j,1}$, $f_{j,2}$, $g_{j,2}$ は実数で, $f_{j,1}$, $g_{j,1}$ は実部, $f_{j,2}$, $g_{j,2}$ は虚部を表し, $\boldsymbol{i}$ は虚数単位を表す. また, $\tilde{F}(x)$, $\tilde{G}(x)$, $A(x)$, $B(x)$ も同様に, それぞれ

$$\begin{aligned} \tilde{F}(x)&=(\tilde{f}_{m,1}+\tilde{f}_{m,2}\boldsymbol{i})x^m+\cdots+(\tilde{f}_{0,1}+\tilde{f}_{0,2}\boldsymbol{i})x^0,\\ \tilde{G}(x)&=(\tilde{g}_{n,1}+\tilde{g}_{n,2}\boldsymbol{i})x^n+\cdots+(\tilde{g}_0x^0+\tilde{g}_{0,2}\boldsymbol{i})x^0,\\ A(x)&=(a_{n-d,1}+a_{n-d,2}\boldsymbol{i})x^{n-d}+\cdots+(a_{0,1}+a_{0,2}\boldsymbol{i})x^0,\\ B(x)&=(b_{m-d,1}+b_{m-d,2}\boldsymbol{i})x^{m-d}+\cdots+(b_{0,1}+b_{0,2}\boldsymbol{i})x^0, \end{aligned} \tag{6}$$

で表す. 上と同様, $\tilde{f}_{j,1}$, $\tilde{f}_{j,2}$, $\tilde{g}_{j,1}$, $\tilde{g}_{j,2}$, $a_{j,1}$, $a_{j,2}$, $b_{j,1}$, $b_{j,2}$ は実数である.

目的関数の導出を行うと, $\|\Delta F\|_2^2+\|\Delta G\|_2^2$ は

$$\sum_{j=0}^{m}[(\tilde{f}_{j,1}-f_{j,1})^2+(\tilde{f}_{j,2}-f_{j,2})^2]+\sum_{j=0}^{n}[(\tilde{g}_{j,1}-g_{j,1})^2+(\tilde{g}_{j,2}-g_{j,2})^2]. \tag{7}$$

となる.

一方, 制約条件の導出について, 式 (4) は

$$\begin{pmatrix} \tilde{f}_{m,1}+\tilde{f}_{m,2}\boldsymbol{i} & & & \tilde{g}_{n,1}+\tilde{g}_{n,2}\boldsymbol{i} & & \\ \vdots & \ddots & & \vdots & \ddots & \\ \tilde{f}_{0,1}+\tilde{f}_{0,2}\boldsymbol{i} & & \tilde{f}_{m,1}+\tilde{f}_{m,2}\boldsymbol{i} & \tilde{g}_{0,1}+\tilde{g}_{0,2}\boldsymbol{i} & & \tilde{g}_{n,1}+\tilde{g}_{n,2}\boldsymbol{i} \\ & \ddots & \vdots & & \ddots & \vdots \\ & & \tilde{f}_{0,1}+\tilde{f}_{0,2}\boldsymbol{i} & & & \tilde{g}_{0,1}+\tilde{g}_{0,2}\boldsymbol{i} \end{pmatrix}$$

$$\times\,{}^t\left(a_{n-d,1}+a_{n-d,2}\boldsymbol{i},\ldots,a_{0,1}+a_{0,2}\boldsymbol{i},b_{m-d,1}+b_{m-d,2}\boldsymbol{i},\ldots,b_{0,1}+b_{0,2}\boldsymbol{i}\right)=\mathbf{0} \tag{8}$$

となる. この式に現われる行列とベクトルについて, それぞれを実部と虚部に分けると, 式 (8) は

$$(N_1 + N_2\boldsymbol{i})(\boldsymbol{v}_1 + \boldsymbol{v}_2\boldsymbol{i}) = \boldsymbol{0}, \tag{9}$$

ここに

$$N_1 = \begin{pmatrix} \tilde{f}_{m,1} & & & \tilde{g}_{n,1} & & \\ \vdots & \ddots & & \vdots & \ddots & \\ \tilde{f}_{0,1} & & \tilde{f}_{m,1} & \tilde{g}_{0,1} & & \tilde{g}_{n,1} \\ & \ddots & \vdots & & \ddots & \vdots \\ & & \tilde{f}_{0,1} & & & \tilde{g}_{0,1} \end{pmatrix}, \quad N_2 = \begin{pmatrix} \tilde{f}_{m,2} & & & \tilde{g}_{n,2} & & \\ \vdots & \ddots & & \vdots & \ddots & \\ \tilde{f}_{0,2} & & \tilde{f}_{m,2} & \tilde{g}_{0,2} & & \tilde{g}_{n,2} \\ & \ddots & \vdots & & \ddots & \vdots \\ & & \tilde{f}_{0,2} & & & \tilde{g}_{0,2} \end{pmatrix}, \tag{10}$$

$$\boldsymbol{v}_1 = {}^t(a_{n-d,1}, \ldots, a_{0,1}, b_{m-d,1}, \ldots, b_{0,1}), \quad \boldsymbol{v}_2 = {}^t(a_{n-d,2}, \ldots, a_{0,2}, b_{m-d,2}, \ldots, b_{0,2}).$$

となる. ここで, 式 (9) の左辺を

$$(N_1 + N_2\boldsymbol{i})(\boldsymbol{v}_1 + \boldsymbol{v}_2\boldsymbol{i}) = (N_1\boldsymbol{v}_1 - N_2\boldsymbol{v}_2) + \boldsymbol{i}(N_1\boldsymbol{v}_2 + N_2\boldsymbol{v}_1),$$

と展開することにより, 方程式 (9) は次の連立方程式

$$N_1\boldsymbol{v}_1 - N_2\boldsymbol{v}_2 = \boldsymbol{0}, \quad N_1\boldsymbol{v}_2 + N_2\boldsymbol{v}_1 = \boldsymbol{0},$$

すなわち

$$\begin{pmatrix} N_1 & -N_2 \\ N_2 & N_1 \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{pmatrix} = \boldsymbol{0} \tag{11}$$

で表される.

さらに, 実係数の場合と同様, $A(x)$ と $B(x)$ の係数に対し, 新たな制約

$$\begin{aligned} \|A(x)\|_2^2 + \|B(x)\|_2^2 = {} & (a_{n-d,1}^2 + \cdots + a_{0,1}^2) + (b_{m-d,1}^2 + \cdots + b_{0,1}^2) \\ & + (a_{n-d,2}^2 + \cdots + a_{0,2}^2) + (b_{m-d,2}^2 + \cdots + b_{0,2}^2) - 1 = 0 \end{aligned} \tag{12}$$

を設ける. これを方程式 (11) の上に配置することにより, 方程式 (11) は

$$\begin{pmatrix} {}^t\boldsymbol{v}_1 & {}^t\boldsymbol{v}_2 & -1 \\ N_1 & -N_2 & \boldsymbol{0} \\ N_2 & N_1 & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \\ 1 \end{pmatrix} = \boldsymbol{0} \tag{13}$$

となる. ここで, 連立方程式 (13) は, 式 (6) の多項式の各係数を変数にもつ連立方程式で, $2(m+n-d+1)+1$ 個の方程式をもつことに注意せよ. 第 $j$ 行の方程式を $q_j = 0$ とおく.

ここで, これまでの多項式の係数を表す変数

$$\begin{aligned} & (\tilde{f}_{m,1}, \ldots, \tilde{f}_{0,1}, \tilde{g}_{n,1}, \ldots, \tilde{g}_{0,1}, \tilde{f}_{m,2}, \ldots, \tilde{f}_{0,2}, \tilde{g}_{n,2}, \ldots, \tilde{g}_{0,2}, \\ & \qquad a_{n-d,1}, \ldots, a_{0,1}, b_{m-d,1}, \ldots, b_{0,1}, a_{n-d,2}, \ldots, a_{0,2}, b_{m-d,2}, \ldots, b_{0,2}) \end{aligned} \tag{14}$$

を, それぞれ $\boldsymbol{x} = (x_1, \ldots, x_{4(m+n-d+2)})$ に置き換える. すると, 式 (7) および方程式 (13) は, それぞれ

$$\begin{aligned} & f(\boldsymbol{x}) = (x_1 - f_{m,1})^2 + \cdots + (x_{m+1} - f_{0,1})^2 + (x_{m+2} - g_{n,1})^2 + \cdots + (x_{m+n+2} - g_{0,1})^2 \\ & + (x_{m+n+3} - f_{m,2})^2 + \cdots + (x_{2m+n+3} - f_{0,2})^2 + (x_{2m+n+4} - g_{n,2})^2 + \cdots + (x_{2(m+n+2)} - g_{0,2})^2, \end{aligned} \tag{15}$$

$$\boldsymbol{q}(\boldsymbol{x}) = {}^t(q_1(\boldsymbol{x}), \ldots, q_{2(m+n-d+1)+1}(\boldsymbol{x})) = \boldsymbol{0} \tag{16}$$

と表される. 以上により, 本稿で考える近似 GCD の問題は, 以下の制約つき最小化問題に帰着される.

**問題 1**
方程式 (16) ($\boldsymbol{q}(\boldsymbol{x}) = \boldsymbol{0}$) の下で, 式 (15) の $f(\boldsymbol{x})$ を最小化せよ. ∎

# 3 近似GCDの算法

本稿では, 与えられた近似 GCD の問題を制約つき最適化問題 (問題 1) に帰着させたものを, 非線形最適化法の一つである勾配射影法 ([10]: 頭文字が本稿の算法名 GPGCD の発祥) もしくは修正 Newton 法 ([14], [22, 第 4 章]) によって解く (詳細は本著者の論文 [16] を参照). 我々のこれまでの実験 [16, Section 5.1] では, 例題に対し, 両算法とも同様の収束性を示す一方, 修正 Newton 法の方がより効率的である結果が得られている. よって, 以後は修正 Newton 法を最適化問題の解法として取り入れる.

実際に修正 Newton 法をもちいて近似 GCD の問題を解くにあたり, 以下の問題を考慮する必要があるので, 下記の各節において議論する.

1. ヤコビ行列 $J_{\boldsymbol{g}}(\boldsymbol{x})$ の構成, ならびに, 反復計算の過程において, ヤコビ行列 $J_{\boldsymbol{g}}(\boldsymbol{x})$ が full rank である (ランクが $J_{\boldsymbol{g}}(\boldsymbol{x})$ の行数に等しい) こと (第 3.1 節).
2. 反復計算の初期値の設定 (第 3.2 節).
3. 最小化問題を最短ベクトル問題に置き換えること (第 3.3 節).
4. GCD となる多項式と, 摂動項の実際の計算 (第 3.4 節).

## 3.1 ヤコビ行列の表現とランク

複素係数多項式 $P(x) \in \mathbb{C}[x]$ を, 次式で表されるものとする.

$$P(x) = p_n x^n + \cdots + p_0 x^0,$$

このとき, 複素数を要素にもつ $(n+k, k+1)$-行列 $C_k(P)$ を次式の通り定義する.

$$C_k(P) = \underbrace{\begin{pmatrix} p_n & & \\ \vdots & \ddots & \\ p_0 & & p_n \\ & \ddots & \vdots \\ & & p_0 \end{pmatrix}}_{k+1}.$$

式 (6) の余因子 $A(x)$, $B(x)$ に対し, 行列 $C_m(A)$ および $C_n(B)$ を, 各要素の実部/虚部から成る行列の和で以下のように表す.

$$\begin{aligned} C_m(A) &= \begin{pmatrix} a_{n-d,1} & & \\ \vdots & \ddots & \\ a_{0,1} & & a_{n-d,1} \\ & \ddots & \vdots \\ & & a_{0,1} \end{pmatrix} + \boldsymbol{i} \begin{pmatrix} a_{n-d,2} & & \\ \vdots & \ddots & \\ a_{0,2} & & a_{n-d,2} \\ & \ddots & \vdots \\ & & a_{0,2} \end{pmatrix} = C_m(A)_1 + \boldsymbol{i} C_m(A)_2, \\ C_n(B) &= \begin{pmatrix} b_{m-d,1} & & \\ \vdots & \ddots & \\ b_{0,1} & & b_{m-d,1} \\ & \ddots & \vdots \\ & & b_{0,1} \end{pmatrix} + \boldsymbol{i} \begin{pmatrix} b_{m-d,2} & & \\ \vdots & \ddots & \\ b_{0,2} & & b_{m-d,2} \\ & \ddots & \vdots \\ & & b_{0,2} \end{pmatrix} = C_n(B)_1 + \boldsymbol{i} C_n(B)_2, \end{aligned} \tag{17}$$

そして, 行列 $A_1$, $A_2$ を次式で定義する.

$$A_1 = [C_m(A)_1 \; C_n(B)_1], \quad A_2 = [C_m(A)_2 \; C_n(B)_2]. \tag{18}$$

($A_1$, $A_2$ は実 $(m+n-d+1, m+n+2)$-行列であることに注意.) このとき, 制約式 (16) の定義により, ヤコビ行列 $J_{\boldsymbol{q}}(\boldsymbol{x})$ は次式の通り求まる.

$$J_{\boldsymbol{q}}(\boldsymbol{x}) = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & 2 \cdot {}^t\boldsymbol{v}_1 & 2 \cdot {}^t\boldsymbol{v}_2 \\ A_1 & -A_2 & N_1 & -N_2 \\ A_2 & A_1 & N_2 & N_1 \end{pmatrix}. \tag{19}$$

(ここに, $\boldsymbol{x}$ の変数名は式 (14) で置き換える前のものを用いていることに注意. $A_1$, $A_2$ は式 (18), $N_1$, $N_2$, $\boldsymbol{v}_1$, $\boldsymbol{v}_2$ は式 (10) を参照.)

修正 Newton 法の反復計算の過程において, ヤコビ行列 $J_{\boldsymbol{q}}(\boldsymbol{x})$ が full rank であることが保証される必要がある (そうでないと, 探索方向を決定できない). この問題については, 以下の命題が成り立つ.

**命題 1**
$\boldsymbol{x}^* \in V_{\boldsymbol{g}}$ を, 方程式 (16) をみたす許容点とする. このとき, $\boldsymbol{x}^*$ に対応する多項式 $\tilde{F}$, $\tilde{G}$ の GCD が $d$ を超えないならば, ヤコビ行列 $J_{\boldsymbol{g}}(\boldsymbol{x}^*)$ のランクは $J_{\boldsymbol{g}}(\boldsymbol{x}^*)$ の行数に等しい.

**証明** Terui [15] を参照. ∎

命題 1 より, 最適化の探索方向が, 探索先の点に対応する GCD の次数を $d$ より大きくするものでない限り, $J_{\boldsymbol{q}}(\boldsymbol{x})$ が full rank であることが保たれ, 近似 GCD の探索方向を適切に保ちつつ反復計算を続けられることがわかる.

## 3.2 反復計算の初期値の設定

反復計算の開始時において, 初期値 $\boldsymbol{x}_0$ は行列の特異値分解 (SVD) [5] に基づいて与える. 与えられた多項式の係数が複素数の場合は, 式 (11) の行列 $N = \begin{pmatrix} N_1 & -N_2 \\ N_2 & N_1 \end{pmatrix}$ に対し, 特異値分解を以下の通り計算する.

$$\begin{gathered} N = U \Sigma {}^t V, \\ U = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{2(m+n-2d+2)}), \quad \Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_{2(m+n-2d+2)}), \quad V = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{2(m+n-2d+2)}). \end{gathered} \tag{20}$$

ここに, $\boldsymbol{u}_j \in \mathbb{R}^{2(m+n-d+1)}$, $\boldsymbol{v}_j \in \mathbb{R}^{2(m+n-2d+2)}$ であり, $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_{2(m+n-2d+2)})$ は対角行列で, 第 $j$ 番目の対角要素が $\sigma_j$ である. 特異値分解の性質 [5, Theorem 3.3] により, 最小特異値 $\sigma_{2(m+n-2d+2)}$ は, $\mathbb{R}^{2(m+n-2d+2)}$ の単位球面

$$\mathrm{S}^{2(m+n-2d+2)-1} = \{\boldsymbol{x} \in \mathbb{R}^{2(m+n-2d+2)} \mid \|\boldsymbol{x}\|_2 = 1\}$$

を $N$ で写した集合

$$N \cdot \mathrm{S}^{2(m+n-2d+1)-1} = \{N\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{R}^{2(m+n-2d+2)}, \|\boldsymbol{x}\|_2 = 1\}$$

上の点の, 原点からの距離の最小値を与える. 式 (20) より

$$N \cdot \boldsymbol{v}_{2(m+n-2d+2)} = \sigma_{2(m+n-2d+2)} \boldsymbol{u}_{2(m+n-2d+2)}$$

が成り立つので,

$$\boldsymbol{v}_{2(m+n-2d+2)} = {}^t(\bar{a}_{n-d,1}, \ldots, \bar{a}_{0,1}, \bar{b}_{n-d,1}, \ldots, \bar{b}_{0,1}, \bar{a}_{n-d,2}, \ldots, \bar{a}_{0,2}, \bar{b}_{n-d,2}, \ldots, \bar{b}_{0,2})$$

に対し, 多項式 $\bar{A}(x)$ および $\bar{B}(x)$ の係数を

$$\begin{aligned}\bar{A}(x) &= (\bar{a}_{n-d,1} + \bar{a}_{n-d,2}\boldsymbol{i})x^{n-d} + \cdots + (\bar{a}_{0,1} + \bar{a}_{0,2}\boldsymbol{i})x^0, \\ \bar{B}(x) &= (\bar{b}_{m-d,1} + \bar{b}_{m-d,2}\boldsymbol{i})x^{m-d} + \cdots + (\bar{b}_{0,1} + \bar{b}_{0,2}\boldsymbol{i})x^0\end{aligned}$$

と定めると, $\bar{A}(x)$, $\bar{B}(x)$ は, 式 (6) において $A(x) = \bar{A}(x)$, $B(x) = \bar{B}(x)$ とおくことにより, $\|A\|_2^2 + \|B\|_2^2 = 1$ を満たしつつ, $AF + BG$ の最小ノルムを与える.

ゆえに, 初期値として, $F$, $G$, $\bar{A}$, $\bar{B}$ の係数からなるベクトル

$$\begin{aligned}\boldsymbol{x}_0 = (&f_{m,1}, \ldots, f_{0,1}, g_{n,1}, \ldots, g_{0,1}, f_{m,2}, \ldots, f_{0,2}, g_{n,2}, \ldots, g_{0,2}, \\ &\bar{a}_{n-d,1}, \ldots, \bar{a}_{0,1}, \bar{b}_{n-d,1}, \ldots, \bar{b}_{0,1}, \bar{a}_{n-d,2}, \ldots, \bar{a}_{0,2}, \bar{b}_{n-d,2}, \ldots, \bar{b}_{0,2}). \quad (21)\end{aligned}$$

を与え, 反復計算を行う.

## 3.3 最小化問題の最近ベクトル問題への置き換え

我々が扱う最小化問題の目的関数 $f$ に対し, 式 (15) より

$$\begin{aligned}\nabla f(\boldsymbol{x}) = 2 \cdot {}^t(&x_1 - f_{m,1}, \ldots, x_{m+1} - f_{0,1}, x_{m+2} - g_{n,1}, \ldots, x_{m+n+2} - g_{0,1}, \\ &x_{m+n+3} - f_{m,2}, \ldots, x_{2m+n+3} - f_{0,2}, x_{2m+n+4} - g_{n,2}, \ldots, x_{2(m+n+2)} - g_{0,2}, 0, \ldots, 0) \quad (22)\end{aligned}$$

が成り立つ. しかし, この最小化問題は, $(x_1, \ldots, x_{2(m+n+2)})$-座標 ($F(x)$ および $G(x)$ の係数に対応する) に関し, 初期値 $\boldsymbol{x}_0$ からの距離が最小であるような点 $\boldsymbol{x} \in V_{\boldsymbol{q}}$ を探索する問題ととらえることができる. ゆえに, 実係数の場合 (Terui [16]) と同様, 最小化問題の目的関数を $\bar{f}(\boldsymbol{x}) = \frac{1}{2}f(\boldsymbol{x})$ とおき, 制約条件 $\boldsymbol{q}(\boldsymbol{x}) = \boldsymbol{0}$ の下で $\bar{f}(\boldsymbol{x})$ の最小値を求める最小化問題を解く.

### 3.4 GCD, および摂動を加えた多項式の計算

反復計算が適切に収束した後, $\tilde{F}(x)$, $\tilde{G}(x)$, $A(x)$, $B(x)$ の係数を得る. $A(x)$ と $B(x)$ は互いに素であるとする. このとき, $\tilde{F}(x)$ と $\tilde{G}(x)$ の GCD である $H(x)$ の係数を計算する必要がある. $H$ は $\tilde{F}$ を $B$ で除した商, あるいは $\tilde{G}$ を $A$ で除した商として求まるが, 素朴な多項式除算は, 係数に誤差を増大させる恐れがある. そこで, $H$ の係数を最小二乗法 [18] で求め, それから $H$ の係数を用いて $\tilde{F}$ および $\tilde{G}$ の修正を行う.

$\tilde{F}$, $\tilde{G}$, $A$, $B$ は式 (6) の通り表され, $H$ は $H(x) = (h_{d,1} + h_{d,2}\boldsymbol{i})x^d + \cdots + (h_{0,1} + h_{0,2}\boldsymbol{i})x^0$, で表されるものとする. このとき, 方程式 $HB = \tilde{F}$ および $HA = \tilde{G}$ を $H$ について解くにあたり, 連立 1 次方程式

$$C_d(A)\,{}^t(h_{d,1} + h_{d,2}\boldsymbol{i}, \ldots, h_{0,1} + h_{0,2}\boldsymbol{i}) = {}^t(\tilde{g}_{n,1} + \tilde{g}_{n,2}\boldsymbol{i}, \ldots, \tilde{g}_{0,1} + \tilde{g}_{0,2}\boldsymbol{i}), \tag{23}$$

$$C_d(B)\,{}^t(h_{d,1} + h_{d,2}\boldsymbol{i}, \ldots, h_{0,1} + h_{0,2}\boldsymbol{i}) = {}^t(\tilde{f}_{m,1} + \tilde{f}_{m,2}\boldsymbol{i}, \ldots, \tilde{f}_{0,1} + \tilde{f}_{0,2}\boldsymbol{i}), \tag{24}$$

を最小二乗法で解くことを考える. この連立方程式 (23), (24) を以下の通り変換する. 式 (24) に対し, ベクトル, 行列をそれぞれ実部と虚部の和で表すことにより,

$$C_d(B) = B_1 + \boldsymbol{i}B_2,$$

$${}^t(h_{d,1} + h_{d,2}\boldsymbol{i}, \ldots, h_{0,1} + h_{0,2}\boldsymbol{i}) = \boldsymbol{h}_1 + \boldsymbol{i}\boldsymbol{h}_2, \quad {}^t(\tilde{f}_{m,1} + \tilde{f}_{m,2}\boldsymbol{i}, \ldots, \tilde{f}_{0,1} + \tilde{f}_{0,2}\boldsymbol{i}) = \boldsymbol{f}_1 + \boldsymbol{i}\boldsymbol{f}_2$$

を得る. よって, 方程式 (24) は

$$(B_1 + \boldsymbol{i}B_2)(\boldsymbol{h}_1 + \boldsymbol{i}\boldsymbol{h}_2) = (\boldsymbol{f}_1 + \boldsymbol{i}\boldsymbol{f}_2). \tag{25}$$

と表される. 実部と虚部それぞれの方程式に分けることにより, (25) は

$$(B_1\boldsymbol{h}_1 - B_2\boldsymbol{h}_2) = \boldsymbol{f}_1, \quad (B_1\boldsymbol{h}_2 + B_2\boldsymbol{h}_1) = \boldsymbol{f}_2,$$

すなわち

$$\begin{pmatrix} B_1 & -B_2 \\ B_2 & B_1 \end{pmatrix} \begin{pmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix}. \tag{26}$$

と表される. これにより, $H(x)$ の係数は, 方程式 (26) を最小二乗法で解くことにより, 得られる. 方程式 (23) の最小二乗解も同様にして求める.

方程式 (23), (24) に対し, 上記の方法によって求めた最小二乗解を, それぞれ $H_1(x), H_2(x) \in \mathbb{C}[x]$ とおく. このとき, $i = 1, 2$ に対し, 残差

$$r_i = \|\tilde{F} - H_i B\|_2^2 + \|\tilde{G} - H_i A\|_2^2,$$

を計算し, 残差 $r_i$ がより小さくなる $i$ に対し, $H_i(x)$ を求める近似 GCD $H(x)$ とする. 最後に, 定めた $H(x)$ に対し, $\tilde{F}(x)$, $\tilde{G}(x)$ を, それぞれ

$$\tilde{F}(x) = H(x) \cdot B(x), \quad \tilde{G}(x) = H(x) \cdot A(x),$$

によって修正する.

## 4 実験

今回提案した, 複素係数多項式のための GPGCD 算法を, 数式処理システム Maple に実装し, 最小二乗法の一つである STLN (structured total least norm) 法を基にした算法 [8] との比較を行った. 計算の比

較は, 係数をランダムに生成した多項式の組に対し, それらの近似 GCD を計算することで行った. 実験環境は Intel Core2 Duo Mobile Processor T7400 (Apple MacBook "Mid-2007") at 2.16 GHz, RAM 2GB, MacOS X 10.5 である.

実験に用いる入力多項式は, GCD をもつ多項式をランダムに生成し, それらに摂動項を加える形で行った. まず, モニックな $m$ 次多項式 $F_0(x)$, $n$ 次多項式 $G_0(x)$ を, $d$ 次の GCD をもつように生成した. GCD$m-d$ 次の多項式, 残りの因子は, $n-d$ の多項式であり, それぞれの係数は $[-10, 10]$ の範囲で乱数で発生させた浮動小数とする. これらに対し, 摂動項として, $m-1$ 次の多項式 $F_{\mathrm{N}}(x)$ と, $n-1$ 次の多項式 $G_{\mathrm{N}}(x)$ を生成した. 係数の与え方は, $F_0(x)$, $G_0(x)$ の係数の与え方に準ずる. そして, 多項式 $F(x)$, $G(x)$ を, それぞれ

$$F(x) = F_0(x) + \frac{e_F}{\|F_{\mathrm{N}}(x)\|_2} F_{\mathrm{N}}(x), \quad G(x) = G_0(x) + \frac{e_G}{\|G_{\mathrm{N}}(x)\|_2} G_{\mathrm{N}}(x),$$

によって定め, $F$, $G$ に対する摂動項の 2-ノルムの大きさが, それぞれ $e_F$, $e_G$ に等しくなるようにする. 本稿では $e_F = e_G = 0.1$ とした.

本稿の実験では, 比較対象である STLN 法を基にした算法 [8] の実装として, その著者らによる実装を用いた (謝辞を参照). STLN 法を基にした算法では, $\mathbb{C}[x]$ 上で複数個の多項式の近似 GCD を計算するプロシージャ`C_con_mulpoly` を用いた. 実験は, Maple 12 上で `Digits=15` の設定, すなわちハードウェアの (倍精度) 浮動小数演算を用いた. 各実験においては, 100 個の多項式をランダムに生成し, テストを行った. 各算法において, 探索の終了を判定するための条件として, 修正 Newton 法では, 探索方向の方向ベクトルのうち, $\tilde{F}(x)$, $\tilde{G}(x)$, $A(x)$, $B(x)$ (式 (4) を参照) の係数に対応する成分の 2-ノルムが $\varepsilon = 1.0 \times 10^{-8}$ よりも小さくなることとし, `C_con_mulpoly` では, 探索終了のしきい値を $e = 1.0 \times 10^{-8}$ とした.

実験結果は表 1 の通りである. $m$, $n$ はそれぞれ $F$, $G$ の次数を表し, $d$ は近似 GCD の次数を表す. 見出しが "STLN" の列は STLN 法に基づく算法の結果を表し, 見出しが "GPGCD" の列は GPGCD 法の結果を表す. "Error" は近似 GCD を得るために与えられた多項式に加えた摂動 $\|\tilde{F}-F\|_2^2 + \|\tilde{G}-G\|_2^2$ の平均値を表す ("$ae-b$" は $a \times 10^{-b}$ を表す). "#Iterations" は反復回数の平均値を表す. "Time" は計算時間 (秒) の平均値を表す.

実験結果を見ると, ほとんどの実験において, STLN 法, GPGCD 法の両方とも, 同程度の大きさの摂動で近似 GCD を計算している一方, 計算効率に関しては, GPGCD 法の方が STLN 法がより効率よく (10 倍〜30 倍程度) 近似 GCD を計算していることがわかる. なお, 実係数多項式に対する GPGCD 法 [16] の場合と異なり, 複素係数多項式に対する GPGCD 法は, すべての実験問題に対して, 十分小さな収束回数で近似 GCD を計算していることに注意する (STLN 法においては, 実係数多項式, 複素係数多項式とも, すべての実験例において反復計算が収束している).

## 5 まとめ

本稿では, 我々の先行研究の結果 [16] を拡張する形で, 複素係数多項式に対する GPGCD 法を示した.

実験結果によると, 本稿で提案した算法は, 実係数多項式に対する算法 [16] と同様, STLN 法に基づく算法と同程度の摂動で近似 GCD を計算する一方, STLN 法に基づく算法よりも大幅 (最大約 30 倍程度) に効率がよいことを示した. これにより, GPGCD 法は, 入力多項式の次数が小〜中程度ならば, 十分実用的な算法であると思われる.

今後の研究方針としては, 理論的な収束性の解明, 反復計算の中で行う連立 1 次方程式の解法で, 行列の構造を考慮した効率的な算法の検討, より多くの個数の入力多項式に対する算法への拡張等を行いたいと考えている.

| Ex. | $m, n$ | $d$ | Error | | #Iterations | | Time (sec.) | |
|---|---|---|---|---|---|---|---|---|
| | | | STLN | GPGCD | STLN | GPGCD | STLN | GPGCD |
| 1 | 10, 10 | 5 | $3.72e{-3}$ | $3.72e{-3}$ | 4.48 | 4.43 | 1.79 | 0.15 |
| 2 | 20, 20 | 10 | $4.16e{-3}$ | $4.16e{-3}$ | 4.24 | 4.22 | 5.88 | 0.30 |
| 3 | 30, 30 | 15 | $4.33e{-3}$ | $4.33e{-3}$ | 4.54 | 4.48 | 14.29 | 0.58 |
| 4 | 40, 40 | 20 | $4.48e{-3}$ | $4.48e{-3}$ | 4.08 | 4.08 | 24.10 | 0.88 |
| 5 | 50, 50 | 25 | $4.63e{-3}$ | $4.64e{-3}$ | 4.05 | 4.12 | 39.19 | 1.36 |
| 6 | 60, 60 | 30 | $4.61e{-3}$ | $4.61e{-3}$ | 4.02 | 4.06 | 60.48 | 1.96 |
| 7 | 70, 70 | 35 | $4.82e{-3}$ | $4.82e{-3}$ | 3.90 | 4.02 | 84.51 | 2.66 |
| 8 | 80, 80 | 40 | $4.84e{-3}$ | $4.84e{-3}$ | 3.88 | 4.04 | 116.03 | 3.65 |
| 9 | 90, 90 | 45 | $4.79e{-3}$ | $4.79e{-3}$ | 3.85 | 4.01 | 151.27 | 4.66 |
| 10 | 100, 100 | 50 | $4.77e{-3}$ | $4.78e{-3}$ | 3.83 | 4.06 | 199.48 | 6.00 |

表 1: Test results for large sets of polynomials with approximate GCD. See Section 4 for details.

## 謝　辞

We thank Professor Erich Kaltofen for making their implementation for computing approximate GCD available on the Internet and providing experimental results.
本研究の一部は, 科学研究費補助金 (若手研究 (B) "多変数代数方程式のべき級数根解法の研究", 課題番号 19700004, 2007–2009 年度) の補助を受けている.

## 参 考 文 献

[1] B. Beckermann and G. Labahn. A fast and numerically stable Euclidean-like algorithm for detecting relatively prime numerical polynomials. *J. Symbolic Comput.*, Vol. 26, No. 6, pp. 691–714, 1998. Symbolic numeric algebra for polynomials.

[2] Paulina Chin, Robert M. Corless, and George F. Corliss. Optimization strategies for the approximate GCD problem. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation (Rostock)*, pp. 228–235 (electronic), New York, 1998. ACM.

[3] R. M. Corless, P. M. Gianni, B. M. Trager, and S. M. Watt. The singular value decomposition for polynomial systems. In *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, pp. 195–207. ACM, 1995.

[4] Robert M. Corless, Stephen M. Watt, and Lihong Zhi. $QR$ factoring to compute the GCD of univariate approximate polynomials. *IEEE Trans. Signal Process.*, Vol. 52, No. 12, pp. 3394–3402, 2004.

[5] James W. Demmel. *Applied numerical linear algebra.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[6] I. Z. Emiris, A. Galligo, and H. Lombardi. Certified approximate univariate GCDs. *J. Pure Appl. Algebra*, Vol. 117/118, pp. 229–251, 1997. Algorithms for algebra (Eindhoven, 1996).

[7] E. Kaltofen, Z. Yang, and L. Zhi. Structured low rank approximation of a Sylvester matrix. In D. Wang and L. Zhi, editors, *Symbolic-Numeric Computation*, Trends in Mathematics, pp. 69–83. Birkhäuser, 2007.

[8] Erich Kaltofen, Zhengfeng Yang, and Lihong Zhi. Approximate greatest common divisors of several polynomials with linearly constrained coefficients and singular polynomials. In *Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, pp. 169–176, New York, NY, USA, 2006. ACM.

[9] Victor Y. Pan. Computation of approximate polynomial GCDs and an extension. *Inform. and Comput.*, Vol. 167, No. 2, pp. 71–85, 2001.

[10] J. B. Rosen. The gradient projection method for nonlinear programming. II. Nonlinear constraints. *J. Soc. Indust. Appl. Math.*, Vol. 9, pp. 514–532, 1961.

[11] Masaru Sanuki and Tateaki Sasaki. Computing approximate gcds in ill-conditioned cases. In *SNC '07: Proceedings of the 2007 international workshop on Symbolic-numeric computation*, pp. 170–179, New York, NY, USA, 2007. ACM.

[12] T. Sasaki and M-T. Noda. Approximate square-free decomposition and root-finding of ill-conditioned algebraic equations. *J. Inform. Process.*, Vol. 12, No. 2, pp. 159–168, 1989.

[13] A. Schönhage. Quasi-gcd computations. *J. Complexity*, Vol. 1, No. 1, pp. 118–137, 1985.

[14] K. Tanabe. A geometric method in nonlinear programming. *J. Optim. Theory Appl.*, Vol. 30, No. 2, pp. 181–210, 1980.

[15] A. Terui. GPGCD, an iterative method for calculating approximate gcd of univariate polynomials, with the complex coefficients. In *Proceedings of the Joint Conference of ASCM 2009 and MACIS 2009*, pp. 212–221, Vol. 22 of COE Lecture Note, December 2009. Faculty of Mathematics, Kyushu University, Fukuoka, Japan.

[16] A. Terui. An iterative method for calculating approximate GCD of univariate polynomials. In *Proceedings of the 2009 International Symposium on Symbolic and Algebraic Computation*, pp. 351–358, New York, NY, USA, 2009. ACM Press.

[17] Christopher J. Zarowski, Xiaoyan Ma, and Frederick W. Fairman. QR-factorization method for computing the greatest common divisor of polynomials with inexact coefficients. *IEEE Trans. Signal Process.*, Vol. 48, No. 11, pp. 3042–3051, 2000.

[18] Zhonggang Zeng. The approximate GCD of inexact polynomials, Part I: a univariate algorithm (extended abstract). preprint. 8 pages.

[19] L. Zhi. Displacement structure in computing approximate GCD of univariate polynomials. In *Computer mathematics: Proc. Six Asian Symposium on Computer Mathematics (ASCM 2003)*, Vol. 10 of *Lecture Notes Ser. Comput.*, pp. 288–298. World Sci. Publ., River Edge, NJ, 2003.

[20] 佐々木建昭, 加古富士雄. 「近似代数」とは？(特集：数式処理とその周辺) . 数理科学, Vol. 36, No. 11, pp. 8–20, November 1998.

[21] 大迫尚行, 杉浦洋, 鳥居達生. 多項式剰余列の安定な拡張算法. 日本応用数理学会論文誌, Vol. 7, No. 3, pp. 227–255, September 1997.

[22] 藤田宏, 今野浩, 田邉國士. 最適化法. 岩波講座 応用数学 [方法 7]. 岩波書店, 1994.